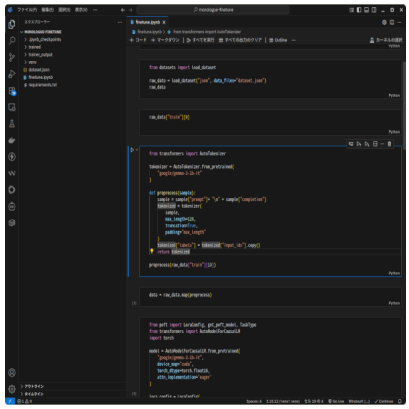


ç?¥è?è??ç©•ã?»å ±æ??ã•®ç ?ç©¶ã??ã••ã•®ï¼? LLMã??ã?¡ã?¤ã?³ã?•ã?¥ã?¼ã??ã?³ã?°ã•®è©¦ã•¿

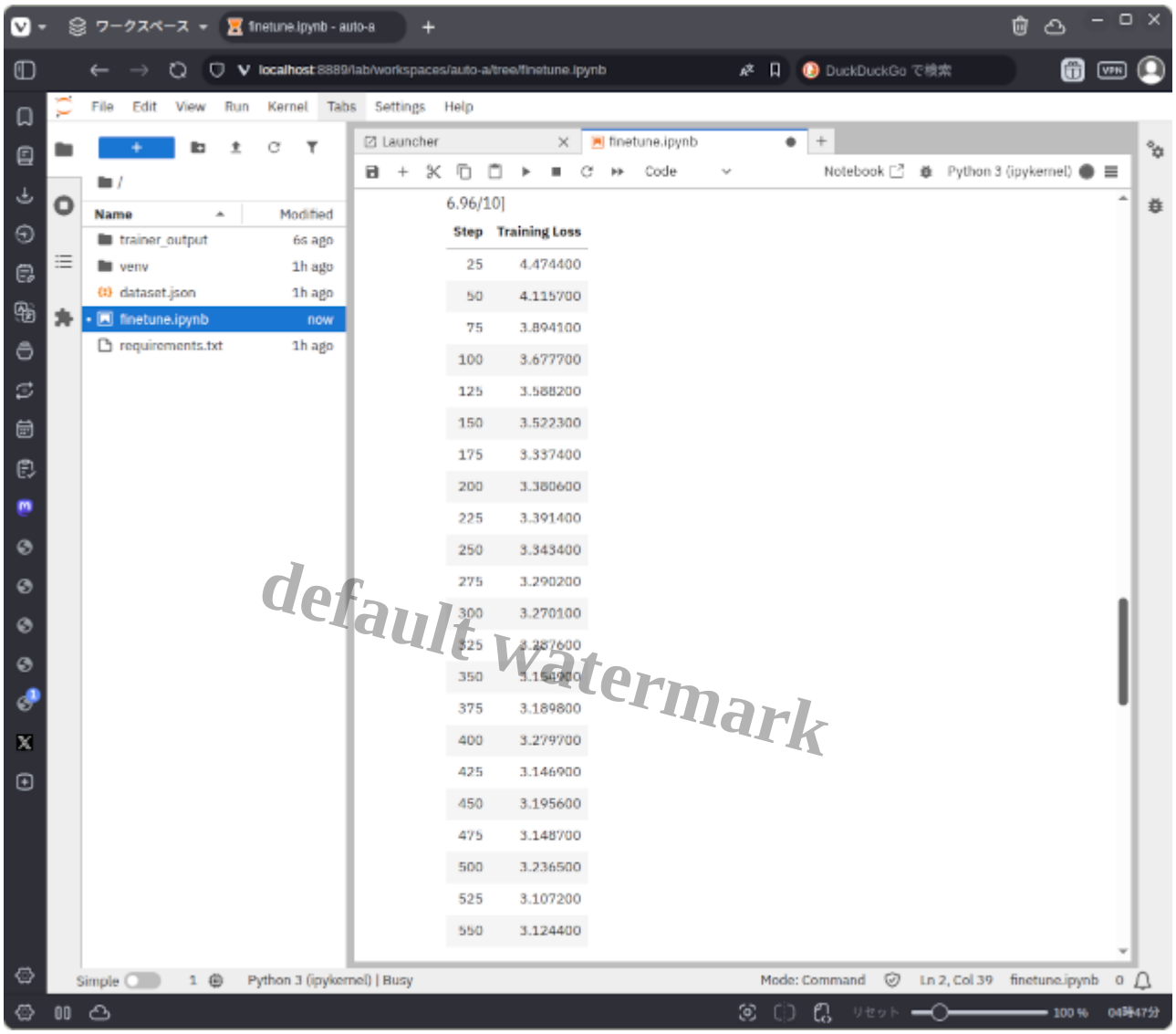
Description

default watermark

å ?æ?¥ã?•ã•?ã•®ã??ã?ã?°ã??ã?¯ã?ã?¼ã?ªã?³ã?°ã?ã•¦æ??ç¨¿ã??æ©?æ¢°å¦ç¿?ç?¨ã•®ã??ã?¼ã?¿ã?»ã??ã??ã•«å¤?æ•?ã•?ã??ã??ã?ã?°ã?©ã? ã??æ?¸ã••ã•¾ã•?ã•?ã??ä»?å??ã•¯ã•ã•®ç¶?ã••ã•§ã?•ã••ã•®ã??ã?¼ã?¿ã?»ã??ã??ã??ä½¿ã•£ã•?æ?¢å?ã•®å-¦ç¿?æ¸?è¨?èª?ã?¢ã??ã?«ï¼?Pretreined Language Modelï¼?ã??Loraã?•ã?¥ã?¼ã??ã?³ã?°ã•?ã??ã??ã?ã?°ã?©ã? ã??æ?¸ã•?ã•¦ã•¿ã•¾ã•?ã•?ã??ï¼?ä¸?ã•®ã?¹ã?¯ã?ªã?¼ã?³ã?·ã?§ã??ã??ï¼?

ä½¿ç?¨ã•?ã•?ã?³ã?³ã??ã?¥ã?¼ã?¿ã?¼ã•¯Core i5 13500 CPUã¨NVIDIA RTX A4000 (16GB VRAN)ã??æ•è¼?ã•?ã•?ã??ã•®ã??å¯¾è±¡ã¨ã•?ã•?è¨?èª?ã?¢ã??ã?«ã•¯Googleã®Gemma 3 1Bã•®ã?¤ã?³ã?¹ã??ã?©ã?¯ã?·ã?§ã?³ã?ã?¥ã?¼ã??ã?³ã?°æ¸?ã•®ã??ã•®ã•§ã•?ã??

çµ•æ??ã•¯å¤±æ??ã??å¦ç¿?ã?ã?¹ã?æ¸?ã??ã•?å®?ç?¨ç??ã•ªã??ã•®ã•«ã•¯ã•ªã??ã•¾ã•?ã??ã•§ã•?ã•?ã??



default watermark

ã•?ã•®ç¨?åº¦ã•®å?¦ç•?ã•§ã•?ã•¨ã?•ã?³ã?³ã??ã?¥ã?¼ã?¿ã?¼ã•®å?¦ç•?é??åº¦ã•«å¤§ã••ã•ªä¸•æº?ã•¯ã•ª ã•?ã??ã•®ã•®ã?•VRAMã•¯ã??ã•£ã•¨æ¬²ã•?ã•?ã•§ã•?ã??ã•?ã•®ã?³ã?³ã??ã?¥ã?¼ã?¿ã?¼ã•®æ§?æ?•ã• §ã•¯1Bã•®ã?¢ã??ã?«ã?•ã??ã?¼ã?¯ã?³é?·128ã•ã??ã•?ã•?é?•ç??ã•§ã?•14GBã•»ã•©ã• ®VRAMã??ä½¿ã•£ã•¦ã•?ã•¾ã•?ã•?ã??

ã?•ã?¤ã??ã?¼ã??ã?©ã?¡ã?¼ã?¿ã?¼ã•¯ä½?åº¦ã??å¤?æ?´ã?ã•¦è©¦ã•?ã•¾ã•?ã•?ã?•è?¯ã?çµ• æ??ã•¯å¾?ã??ã??ã•¾ã•?ã??ã•§ã•?ã•?ã??

ã??ã?¼ã?¿ã?»ã??ã??ã•®è³ªã•?æ?ªã•?ï¼?ã??ã?ã?°æ??ç¨¿ã••ã•®ã•¾ã•¾ã•¯æ??ã•¾ã•?ã••ã•ªã•?ï¼?ã• ¨ã•?ã•?ã•¨ã•?ã??ã•?ã??ã•¾ã•?ã??ã??

å•?è??ã•¾ã•§ã•«ã?•ã??ã?¬ã?¼ã??ã?³ã?°ã?³ã?¼ã??ã•¯æ¬¡ã•®ã??ã•?ã•ªã??ã•®ã•§ã•?ã??

```
/* ã??ã?¼ã?¿ã?»ã??ã??ã??èªã•¿è¾¼ã?? */

from datasets import load_dataset

raw_data = load_dataset("json", data_files="dataset.json")
```

```
/* ã??ã?¼ã?¯ã??ã?¤ã?¶ã?¼ã•®æº?å?? */

from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained(
    "google/gemma-3-1b-it"
)

/* ã??ã?¼ã?¿ã?»ã??ã??ã•®å?•å?¦ç•? */

def preprocess(sample):
    sample = sample["prompt"]+ "\n" + sample["completion"]
    tokenized = tokenizer(
        sample,
        max_length=128,
        truncation=True,
        padding="max_length"
    )
    tokenized["labels"] = tokenized["input_ids"].copy()
    return tokenized

data = raw_data.map(preprocess)

/* ã?•ã?¥ã?¼ã??ã?³ã?°å¯¾è±¡ã•®ã?¢ã??ã?«ã•®æº?å?? */

from peft import LoraConfig, get_peft_model, TaskType
from transformers import AutoModelForCausalLM
import torch

model = AutoModelForCausalLM.from_pretrained(
    "google/gemma-3-1b-it",
    device_map="cuda",
    torch_dtype=torch.float16,
    attn_implementation='eager'
)

/* LORAã•®è¨å®? */
lora_config = LoraConfig(
    task_type=TaskType.CAUSAL_LM,
    target_modules=["q_proj", "k_proj", "v_proj"]
)

model = get_peft_model(model, lora_config)

/* ã??ã?¬ã?¼ã??ã?¼ã•®æº?å?? */

from transformers import TrainingArguments, Trainer

train_args = TrainingArguments(
    num_train_epochs=10,
    learning_rate=3e-4,
    logging_steps=25,
    fp16=True
)
```

default watermark

```
trainer = Trainer(
    args=train_args,
    model=model,
    train_dataset=data["train"]
)

/* ã??ã?¬ã?¼ã??ã?³ã?°å®?è¡? */
trainer.train()

/* ã??ã?¬ã?¼ã??ã?³ã?°æ¸?ã•®ã?¢ã??ã?«ã•¨ã??ã?¼ã?¯ã??ã?¤ã?¶ã?¼ã•®ä¿•å? */

trainer.save_model("./trained")
tokenizer.save_pretrained("./trained")
```

å¦ç¿?ã?ã?¹ã•¯1.0ä»¥ä¸?ã?ç?®æ¨?ã•§ã•?ã•?ã?•3.0ã•¾ã•§ã•?ã•?æ¸?ã??ã•¾ã•?ã??ã•§ã•?ã•?ã??

ä»¥ä¸?ã•®ã?³ã?¼ã??ã•?ã??ã?¡ã?¤ã?³ã?•ã?¥ã?¼ã??ã?³ã?°å¾?ã•®ã?¢ã??ã?«ã??ä½¿ã•£ã•?æ?¨è«?ã??è¡?ã•?ã??ã•®ã•§ã•?ã??

```
from transformers import pipeline

ask_llm = pipeline(
    model="./trained",
    tokenizer="./trained",
    device="cuda"
)

print(ask_llm("ã??ã?ã?³ã??ã??")[0]["generated_text"])
```

ä»?å??ã•®ã•¨ã•?ã?•ã•¯ã?•æ??æ³?ã•?ç•?è§£ã•§ã••ã•?ã•¨ã•?ã•?ã•?ã•¨ã•?å•?ç©«ã•¨ã•?ã•?ã•?ã•¨ã•«ã•?ã•¦ã?•èª²é¡?ã??æ?´ç•?ã•?ã•¦ã•?ã??å?•æ??æ?¦ã•?ã•¾ã•?ã??

default watermark

Category

1. æ?¥ã? ã•®æ´»å??

Tags

1. AI
2. Linux
3. LLM
4. Lora
5. ã??ã?¡ã?¤ã?³ã?•ã?¥ã?¼ã??ã?³ã?°
6. ä¸?é??å??å¸?
7. å¤§è¦•æ¨¡è¨?èª?ã?¢ã??ã?«
8. å±±æ¢¨ç??

Date Created
2025å¹´8æ??3æ?¥
Author
kazuo-tsubaki